The PCA (Principal Components Analysis) Program

Matthew Marcus
August, 2005
Update: August, 2010

I. Introduction

PCA is a method for analyzing a set of spectra to see if they can be represented as linear combinations of a smaller number of component spectra. This program implements PCA as described by Ressler (*Environ. Sci. Technol.*(2000) **34**,950-958). The notation in the program is as used in that paper.

II. Main screen

The program starts out by asking for unknowns, one at a time. By default, files with extension `e`, `b`, `bk3` or `pca` are shown in the file dialog. To end, hit Cancel in the file-open dialog box. After that, you see a screen which looks like that in Figure 1, except for the red labels.

You can also specify a database file (`*.prm`) of the same sort as is used in the Combination Fit program. Such a file can look like this:

```
#Ref=Greenrust_SO4_borch.e
#Ref=fe_cr2feo4_xafs_tr_nico.e
Ref=Augite FeXANES ITFA comp0.e
Ref=Augite FeXANES ITFA comp1.e
Ref=Augite FeXANES ITFA comp2.e
Ref=Augite FeXANES ITFA comp3.e
Ref = ""
```

with everything after a # ignored. This option is handy when dealing with large numbers of files.

This screen shot was taken with a demonstration dataset, each file of which was made by taking linear combinations of the same three references (Fe foil, Fe_3O_4 and 2-line ferrihydrite) and then adding a small amount of noise. This is therefore a highly-idealized case in which PCA will tell us what we already know, that there are three components.

At the upper left is the list of paths you selected. To the right of this is a set of checkboxes wired like radio buttons so you can only click one at a time. This selects which unknown is plotted, along with its reconstruction from the chosen set of

components and the residual thereof. These data are shown on the graph to the right in white for the input data, green for the reconstruction and red for the residual.

Above the graph are four 'badness-of-fit' indicators. Two of these are for the individual file shown in the graph, and the others are the average over the whole set. These quantities are defined as $\sum (y_i - y_i^{fit})^2 / \sum y_i^2$ and $\sum \left| y_i - y_i^{fit} \right| / \sum \left| y_i \right|$, with the index ranging over the points in an individual curve or the whole set.

The component list to the lower right shows the breakdown into the abstract components. Note that these components are at best linear combinations of spectra corresponding to species in the unknowns and often don't look much like EXAFS or XANES. The first column in the list is the eigenvalues, whose squares represent the contribution to the data made by that particular component. Next is the indicator (*IND*) value of Malinowski, a measure of the usefulness of adding in another component. According to the semi-empirical theory of errors in PCA, the last useful component is the one at which *IND* is a minimum. The next column is a set of checkboxes showing which components are in the fit, according to the setting of the control to the left. If you add all of them, as is the case when the program starts, the fit is perfect and perfectly meaningless. Here, we've added the first three. Next is another column of checkboxes allowing you to select one component to be plotted on the graph at lower right. This component is weighted by the eigenvalue, so insignificant components come out small. The contribution of each weighted component to the data being plotted in the upper graph is given by the column of numbers at the right edge of the components list complex. If a component is not selected for inclusion, its contribution to the whole is zero, regardless of the number in the contributions column. You can save each individual component to a file (default extension `cmp`). If you do a linear least-squares fit to one of the data files using these components as references, you will get the coefficients shown in the contributions column. Thus, in the present example, the first data file (f1-0-0.e) fits to 0.307**comp0*-0.498**comp1*+0.449**comp2*, where comp*i* is the i^{th} component.

The residual of the fit is shown in red on the top graph. You can multiply it by a factor given by the setting of the Residual Magnification control, in this example, 10x. The cursor in the bottom (components) graph can be locked in X motion to that in the top

graph, which can be useful for seeing which features on the data correlate with which ones in the components.

If you want to add a file to or drop one from the list, you can use the green buttons associated with the file list. Use the index spinner to put at the top of the indicator the file you want to drop. An added file goes on the bottom. In any case, the number of components gets reset to the current number of spectra.

It should be noted that the method rarely works as well on real data as it does on this simulated data. Don't expect the residual to look like pure white noise even when you add the right number of components. In this example, the fourth and higher components all look like pure noise, while there's obvious signal in the first three (not shown). For real data, it's not so obvious and one must look to the *IND* value and prior knowledge about what's reasonable for the system.

The matrix of loadings (weights) may be saved by pushing the Save Amounts button. This saves a tab-delimited text file whose first line is a list of the filenames of the spectra. Succeeding lines give the weights for all the spectra in component 0, component 1, etc. The fits and components may also be saved by pressing the appropriate buttons.

III. Target transformation

The PCA fit gives no indication of what the components actually represent. One way of finding out is to use the target transformation. This procedure takes a reference data file and removes from it everything which doesn't look like something found in the unknowns. This is done by making a linear least-squares fit of the reference to a weighted sum of components. Thus, if the unknowns can be represented as mixtures containing the reference being tested, the target transformation will leave the reference spectrum unaffected. Otherwise, the output won't look like the input. The screen in which this test is done is shown in Figure 2.

In this case, we have tested a reference which was not one of the three from which the simulated data were made. The *SPOIL* value (a measure of how much the target transformation disagrees with the input) is much greater than the 0-3 one expects to see for a reference which is really represented in the data, and the reconstruction looks nothing like the original. In most real cases, it's not so obvious that a reference doesn't

belong, which is why the *SPOIL* value is computed. There are buttons which allow you to save the transformed file (default extension `trg`) and to read in another candidate reference.

It is assumed that the test reference data covers the same or almost the same range as the data from the unknowns. There is a method for doing the target test which lets one violate this assumption, but this program doesn't do it.

IV. Rotations

The abstract components are usually not very informative about what's going on physically. The weights given to spectra usually include lots of negative numbers. When the data really are describable as a sum of signals from actual species, the abstract components end up being weighted sums and differences of the spectra from the species. For instance, if there are two components, the top one is an average and the bottom one a difference between species. Thus, it's often useful to make some sort of intelligent choice of linear combinations. One of these is the varimax rotation [Kaiser, *Psychometrika* **23**,187 (1958)]. This is an orthogonal rotation of the components so that the unknown spectra get assigned values for the weights which are as uncorrelated with each other, so that the information in Unknown #1 isn't repeated in the description of Unknown #2. The varimax rotation maintains the orthonormality of the component basis, so that $\sum_{p=1}^{\#points} U_{pc}U_{pc'} = \Lambda_c^2 d_{cc'}$ where U_{pc} is the value of component c at point p and Λ_c is the eigenvalue for component c. This rotation is an alternate way of viewing the results and can be useful in classifying spectra into groups based on similarities in weightings.

A next step is Iterative Target Factor Analysis [see Scheinost, et. al., *Physica Scripta* **T115**, 1038(2005) and Rossberg, et. al, *Anal. Bioanal. Chem.* **376**, 631(2003)], in which one tries to extract component spectra with some physical meaning. The idea is to make a new basis set by doing linear combinations of the old so that each unknown is described as a sum of the new components with coefficients (weights) between 0 and 1. The starting point for this is the varimax solution. This operation is not strictly a rotation because the orthonormality described above no longer holds. For instance, XANES spectra all go from 0 at one end to 1 (if normalized) at the other, so the scalar product of

two physical spectra can't be 0. In any case, the result of ITFA is a set of spectra which often look very much like the underlying physical species, and a set of weighting factors which range between 0 and 1. In the highly-idealized data used as an example, the actual spectra and recovered components are shown in Figure 3.

For some data sets, there is a presumption that the amounts of the components (loadings) should add up to 1 for each input spectrum. The ITFA adj for sum=1 view rescales the ITFA components to make that happen, as nearly as possible. Each ITFA component is multiplied by a scale factor chosen to minimize the mean-squared deviation of the sums of the loadings from 1. For post-edge normalized XANES data (*.e files), this procedure makes the edge jump of the ITFA components, which look like XANES spectra, equal to 1.

These alternate views may be selected by using the View switch on the left side of the component-selection area. The displayed loadings and components change according to which view is selected. Note that for the Varimax and ITFA views, the components depend on the number displayed. The underlying eigenvectors seen in the Abstract view don't change, but the weighted sums you get as components do.

There is a version of ITFA in which one can do what is essentially target transformation on a reference whose data range does not include all the range available for the unknowns, the result of which is that you 'fill in' the missing data from the information in the unknowns. This program does not perform this version at this time.

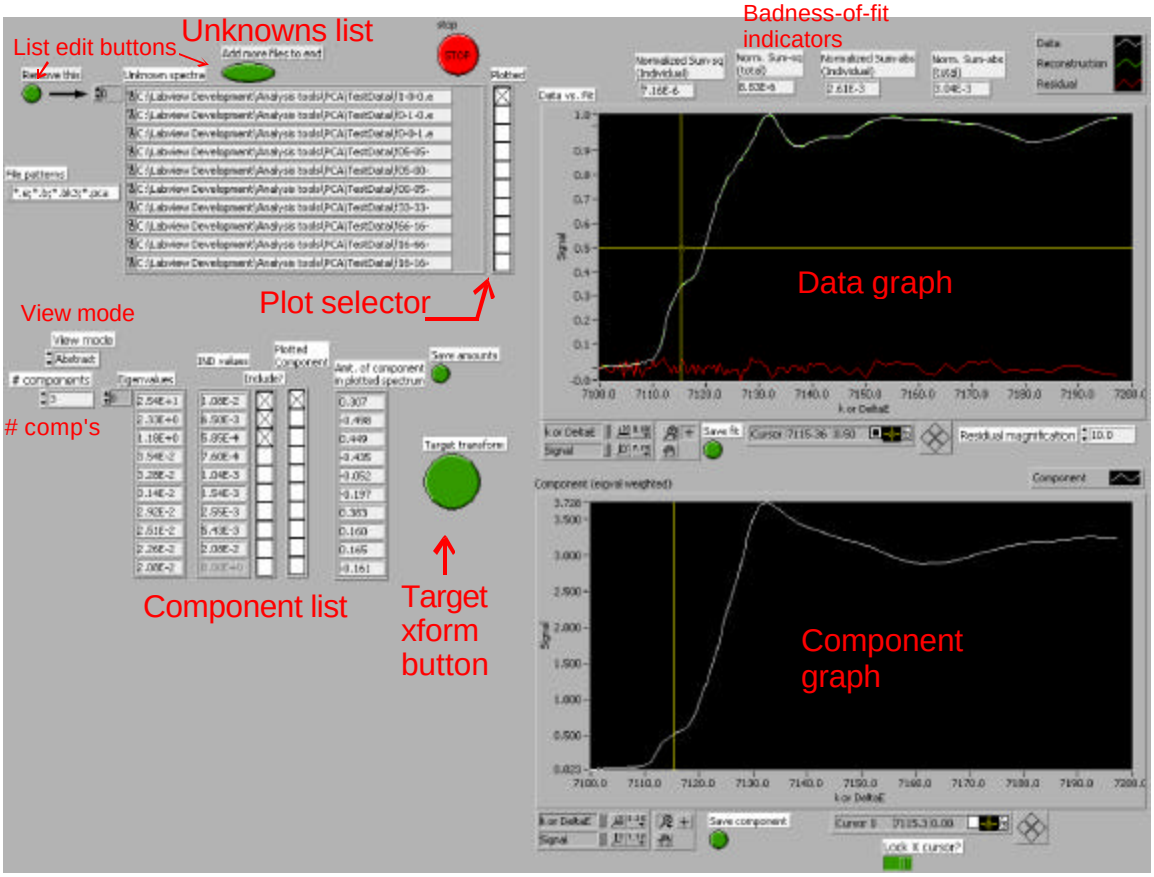

Figure 1. The main PCA screen, for a demonstration set of data. The fit has been limited to the first three components, the second one of which is shown in the component graph. The first data file is plotted along with a fit and the resulting residual, which is magnified 10x.

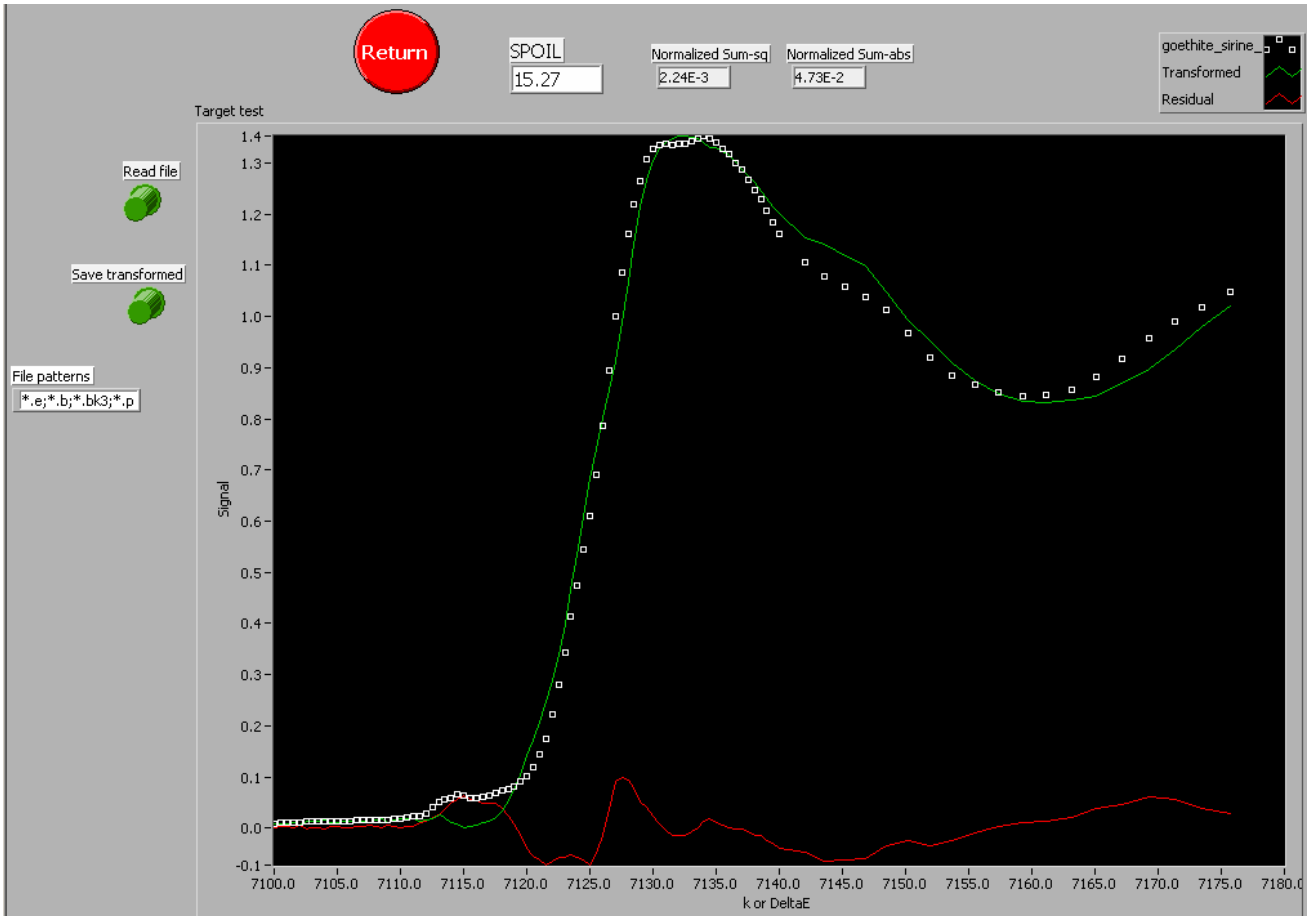

Figure 2. The target-transformation screen. The reference being tested is not one of the ones from which the demonstration data set was created. Therefore, the SPOIL value is high and the transformed data don't resemble the input.

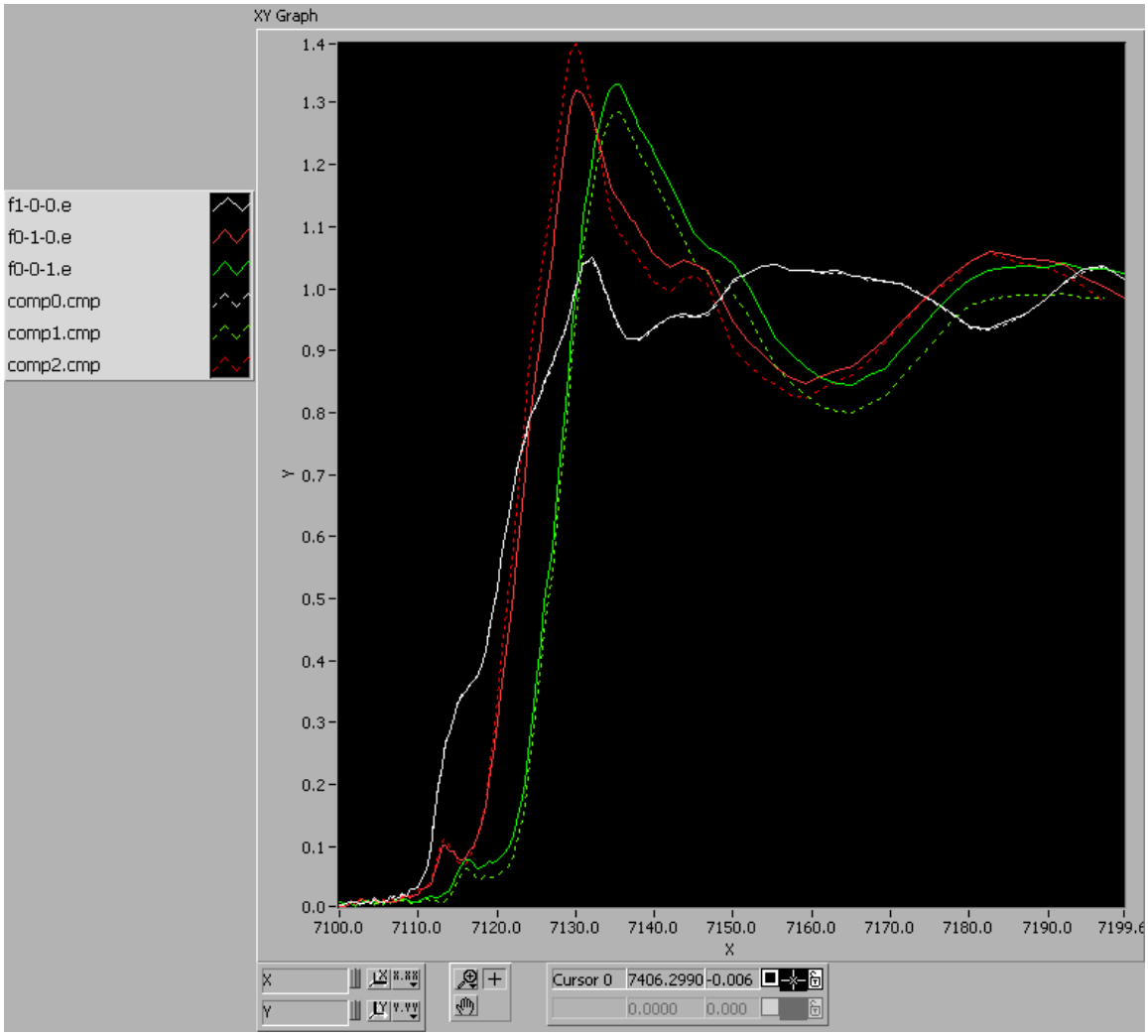

Figure 3. The actual spectra (solid lines) for the test data set and the components recovered by ITFA (dashed). The components are Fe foil (white), Fe_3O_4 (red) and ferrihydrite (green).